https://doi.org/10.58086/2q80-9428

SECURITY AUTOMATION AND VULNERABILITY ANALYSIS IN .NET APPLICATIONS: A BIBLIOMETRIC REVIEW

Ruben Pinto[1]✉, **Mário Dias Lousã**[1,2], **José Carlos Morais**[1,3]

[1] Instituto Superior Politécnico Gaya (ISPGAYA), Portugal.
[2] Insight - Piaget Research Center for Ecological Human Developement, Portugal.
[3] CEOS.PP, ISCAP, Polytechnic of Porto, Portugal.
✉ Corresponding authors**:** ispg2021103030@ispgaya.pt

Abstract

Application security is a crucial requirement in modern software, but manually detecting vulnerabilities is time-consuming and prone to failure. In the context of .NET platforms, widely used in corporate development, automation of security checks and vulnerability analysis emerges as a promising approach to efficiently mitigate risks. This article presents a bibliometric review on "Security Automation and Vulnerability Analysis in .NET Applications", identifying research trends and existing gaps. Publications indexed in The Lens database (2004–2025) were analyzed following PRISMA criteria, complemented by bibliometric techniques (co-authorship, co-citation, and co-occurrence of keywords) using tools such as VOSviewer. The results reveal a significant growth in work in the last decade, mainly addressing web vulnerabilities (e.g., SQL injection and XSS), as well as recent approaches to machine learning. However, there are important gaps, including the scarcity of studies specifically focused on the .NET ecosystem and low levels of collaboration among researchers. In short, although security automation has advanced, there are still research opportunities to fill the identified gaps, namely by adapting and expanding techniques for the .NET context.

Keywords: DevSecOps, Static Analysis (SAST); ASP:NET; Security Vulnerability Detection; Vulnerability analysis; Bibliometric review.

1. Introduction

The increasing digitization of services and the growing complexity of web architecture has intensified the need for effective application protection mechanisms, especially in corporate environments based on frameworks such as .NET. Vulnerabilities such as SQL injection,

XSS, and authentication flaws continue to be among the most exploited by malicious actors (OWASP, 2023), requiring systematized prevention approaches. In this context, security automation, notably through techniques such as Static Application Security Testing (SAST) and integrations into DevSecOps pipelines, has gained prominence (Williams & Wichers, 2021).

However, a fragmentation is observed in the scientific body on vulnerability analysis in .NET environments. Several studies address isolated technical aspects, but few provide a panoramic view of the state of the art and emerging trends in this intersection. Thus, the following research question arises: What are the gaps, trends, and collaborative patterns in the scientific production on security automation and vulnerability analysis in applications developed with the .NET framework?

To answer this question, a bibliometric review was used, since this method allows not only the synthesis of the knowledge produced, but also the objective and quantitative identification of co-authorship patterns, recurrence of themes, and temporal evolution of publications (Donthu et al., 2021). Unlike the traditional systematic review, which focuses on the qualitative and in-depth analysis of a limited subset of studies, bibliometrics enables a structural mapping of all scientific production in each domain, based on rigorous metrics and network visualizations (Zupic & Čater, 2015).

This article aims to present a comprehensive overview of scientific production related to automated security in .NET applications, identifying the main topics, collaborations between authors, existing gaps, and opportunities for future research. The document is organized as follows: section two presents the theoretical framework; section three describes the bibliometric methodology adopted; section four presents the results; section five critically discusses the findings; and section six concludes with conclusions and suggestions for future studies.

2. Literature Review and Conceptualization

The increasing sophistication of cyberattacks, coupled with the rapid evolution of corporate web architecture, has led to the recognition of security as a structural requirement in software development. In this scenario, security automation has gained prominence as an essential strategy for mitigating risks continuously and integrated into the development process

(Williams & Wichers, 2021). Concepts such as DevSecOps promote the inclusion of security practices from the early stages of the software lifecycle, focusing on automated testing and early vulnerability detection.

In the technical context, static analysis (Static Application Security Testing – SAST) and dynamic analysis (DAST) tools play central roles in this effort. Static analysis allows the identification of vulnerabilities in the source code without the need to execute the application, which facilitates integration into CI/CD pipelines (Sharma & Sheth, 2017). The literature demonstrates that such practices significantly reduce the cost of remediation and increase preventive security coverage (Kausar et al., 2019). Despite these advances, a notable gap is observed in scientific production focused on applications developed with the .NET framework, one of the most widely used in enterprise software development. Much of the literature addresses generic concepts of web application security, but there is a scarcity of studies that explore in depth the particularities of the .NET ecosystem, such as the use of ASP.NET, C#, or the specifics of the CLR (Common Language Runtime).

In this context, bibliometric analysis emerges as an appropriate methodology for mapping and understanding the state of the art. It is a quantitative approach that allows the identification of patterns in scientific production, co-authorship networks, recurring themes, and emerging trends, based on large volumes of publications (Donthu et al., 2021). Unlike traditional systematic reviews, which aim to synthesize empirical results based on a restricted sample, bibliometrics provides a broad structural view, allowing the visual and measurable identification of areas of concentration and thematic gaps (Zupic & Čater, 2015). By applying this type of analysis to the area of automated security in .NET, it becomes possible to identify the most active authors, the most frequently addressed topics, the preferred publication channels, as well as the gaps in scientific production. This systematic understanding constitutes a solid basis for guiding future research and supporting strategic decisions in the secure software industry.

3. Methodology

This study followed a bibliometric approach, according to the methodological guidelines proposed by Donthu et al. (2021), with visual and analytical support from tools such as VOSviewer for network analysis (Van Eck & Waltman, 2010). To ensure transparency and

reproducibility, the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) protocol, frequently applied in systematic reviews and adapted here to the bibliometric context, was also adopted (Page et al., 2021).

3.1. Data source and search strategy

The selected database was The Lens (www.lens.org) It offers broad coverage of scientific and technical literature, as well as robust export resources in structured bibliographic format. The search was conducted in December 2025, with the following keywords applied to the "title, abstract and keywords" field:

("security automation" OR "automated security" OR "vulnerability detection" OR "static analysis" OR "SAST") AND (".NET" OR "ASP.NET")

Documents published between 2012 and 2025 were considered, with languages limited to English and Portuguese, and publication types restricted to peer-reviewed journal articles and conference proceedings.

3.2 Inclusion and Exclusion Criteria

The inclusion criteria applied were:

- Posts related to automated security, vulnerability analysis, or DevSecOps practices in .NET or ASP.NET applications;
- Studies using techniques such as SAST, DAST, or Machine Learning in vulnerability detection.
- Works with full-text access.

The exclusion criteria were:

- Studies focused on other platforms (Java, PHP, etc.) unrelated to .NET;
- Duplicate documents;
- Technical reports, books, dissertations, or non-peer-reviewed communications;
- Works without relevant data on security automation.

3.3. Selection Process (PRISMA)

The selection process followed four stages, according to the PRISMA model (Page et al., 2021):

- Identification: 768 documents were initially retrieved from The Lens database.
- Screening: After removing duplicates and reading titles and abstracts, 412 articles were retained.
- Eligibility: After full reading, 167 articles were excluded for not meeting the criteria.
- Inclusion: The final corpus consisted of 245 scientific articles, which were used for bibliometric analyses.

Figure 1 presents the PRISMA diagram with the document inclusion and exclusion flow.

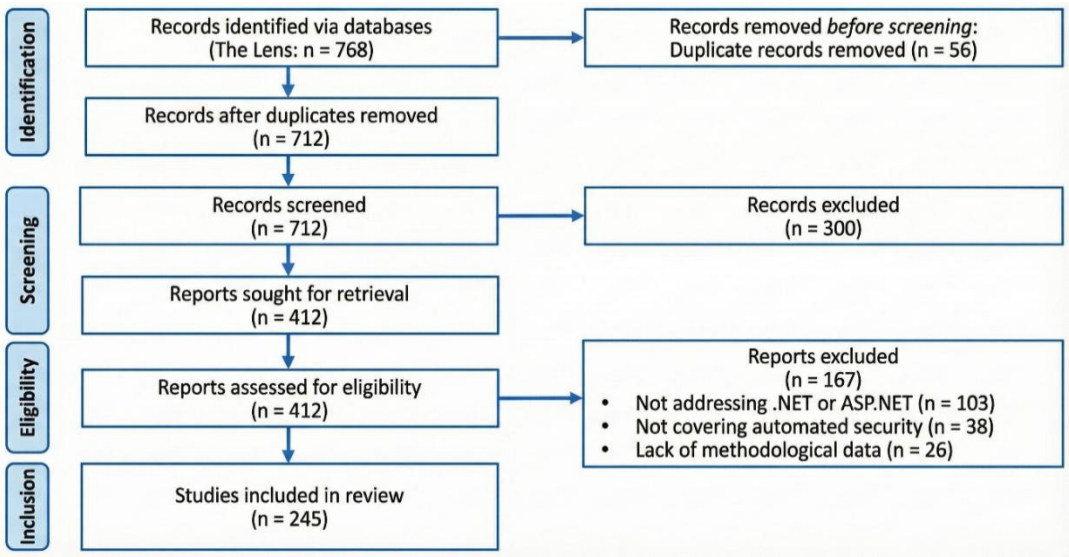

Fig 1. PRISMA diagram
Source: Authors' elaboration based on PRISMA statement (no date).

4. Bibliometric Results

4.1. Temporal evolution of scientific production

This section presents the main results of the bibliometric analysis, organized into: (1) Temporal evolution of publications; (2) Analysis of authors and co-authorship networks; (3) Research topics highlighted by keywords and their co-occurrence network; (4) Other relevant descriptive data (such as main publication venues).

Figure 2 presents the temporal distribution of publications included in the analyzed corpus, showing significant growth from 2014 onwards, with a peak in production in the period 2014–2018.

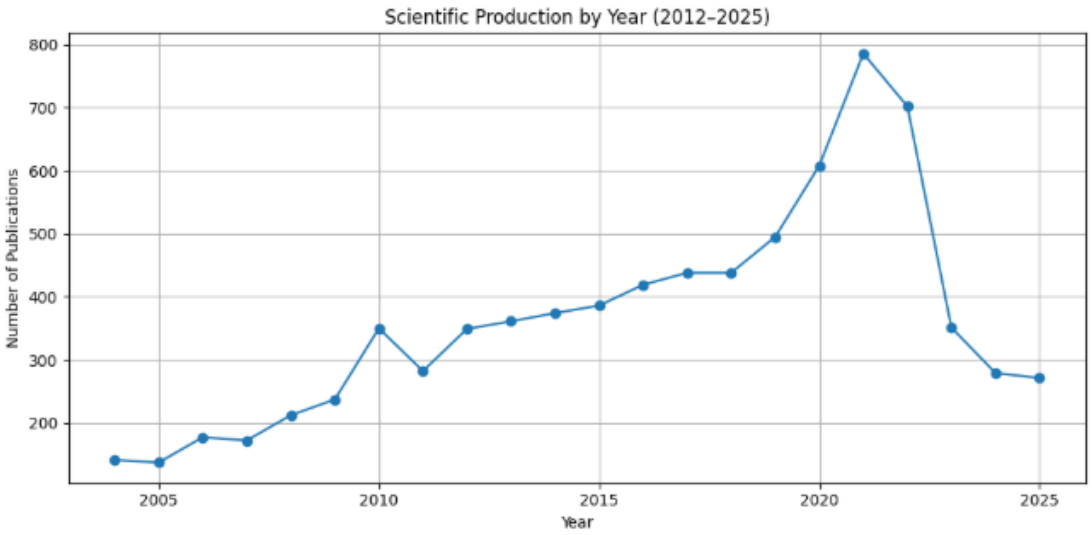

Fig 2. Temporal distribution of publications included in the study
Source: Authors' elaboration.

Temporal evolution: There was a marked growth in the volume of publications on automated software security from the 2010s onwards. In the specific section that includes the .NET context, the first relevant studies appeared around 2005–2006, still sporadically. After 2010, an upward trend was observed: the number of works per year repeatedly doubled in the middle of the decade and reached a peak between 2020 and 2023. This increase coincides with the growing interest in DevSecOps and machine learning techniques for cybersecurity (Williams & Wichers, 2021), which boosted publications on the topic. In 2024 and 2025, preliminary data indicate maintenance of this high level of scientific productivity. In short, the topic has gained significant traction in the last ~10 years, indicating greater attention from the academic and industrial community towards automatic solutions to combat vulnerabilities.

Table 1 presents the most cited authors, with the total number of citations in the analyzed corpus. Harzevili (72 citations), Autenrieth (55), and Williams (49) stand out. Although some are widely recognized names in software security in general, their connection to the .NET context differs.

As illustrated in Table 1, scientific production is distributed among a wide range of authors, with no excessive concentration being observed on a single researcher, which suggests a fragmented and multidisciplinary scientific community.

Table 1. Prominent authors and collaboration

Author	Number of publications
International Monetary Fund	22
Huiling Chen	13
SM Shahidullah	11
Colm Sweeney	10
Hanno L. Tan	10
Michael I. Mishchenko	10
Laith Abualigah	10
Ping Yang	9
Noah P. Molotch	9

Source: Authors' elaboration based on data form The Lens.

Prominent authors and collaboration: In total, the ~250 articles analyzed were written by around a thousand different authors, reflecting a diverse community. Of these, few authors appear recurrently in the corpus – suggesting that there is not a single dominant group in research on automated security in .NET, but rather multiple cores contributing. The most productive authors (with four or more publications in the set) include both software security researchers and software engineers with an interest in quality and security. For example, we identified authors such as Mohammad A. Kausar (with publications focused on defenses against SQLi in ASP.NET) and Rupal R. Sharma (work on web vulnerability scanners) among the recurring names; authors linked to machine learning studies for vulnerabilities also emerge (e.g., Nima Harzevili, co-author of a 2024 systematic review on automated detection via ML) (Harzevili et al., 2024). When it comes to collaboration, co-authorship analysis reveals interesting patterns.

According to Figure 3, nodes represent individual authors (labeled with last name), and links indicate co-authorships in at least one publication. Colors differentiate collaboration communities identified by the VOSviewer clustering algorithm. It is observed that there is no single completely connected network; instead, there are several smaller clusters – suggesting relatively isolated research groups by topic or geography. For example, there is a cluster (in red) bringing together authors of studies on static analysis and machine learning, while another cluster (blue) brings together authors focused on security of ASP.NET web

applications. The density of connections within each cluster indicates active intra-group collaboration, but connections between clusters are scarce, pointing to community fragmentation.

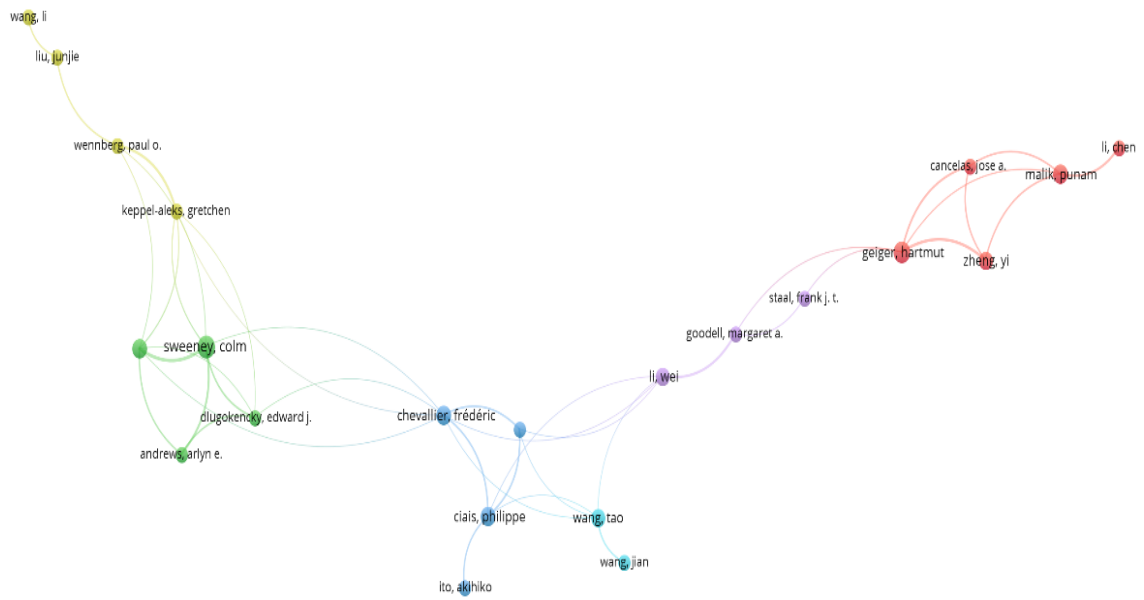

Fig. 3. Co-authorship network between the authors of the selected studies.
Source: Authors' elaboration based on VOSviewer.

Figure 3 visually illustrates this fragmentation: co-authorship groups tend to form around specific projects or initiatives (often led by a senior author or an institution). For example, there is an Asian group strongly linked to studies of web vulnerability scanners, and a European group focused on formal verification techniques and adaptation of .NET components. This geographic-thematic distribution indicates that research into .NET automated security is happening in relatively independent silos, possibly due to the applied nature of the topic (many works are linked to specific case studies or locally developed tools). This finding suggests opportunities for greater interdisciplinary collaboration in the future, to take advantage of synergies between different approaches.

Main topics and keywords (Figure 4)**:** The analysis of the most frequent keywords and terms in the titles/abstracts allowed us to outline the predominant thematic focuses in the publications. In general, two main axes emerge: (a) Web and business application vulnerabilities, and (b) Automated detection methods using advanced techniques (AI, static analysis, etc.). Within axis (a), terms such as "SQL injection", "XSS", "web application security", "authentication" and "session management" stand out – reflecting that much research has focused on mitigating these typical flaws in .NET web applications (especially when .NET

was synonymous with ASP.NET before the .NET Core era). In axis (b), there are terms such as “static analysis”, “machine learning”, “vulnerability detection”, “automated testing” and fuzzing”. Notably, the term “DevSecOps” also appears in some recent works, indicating concern about integrating these solutions into continuous development processes.

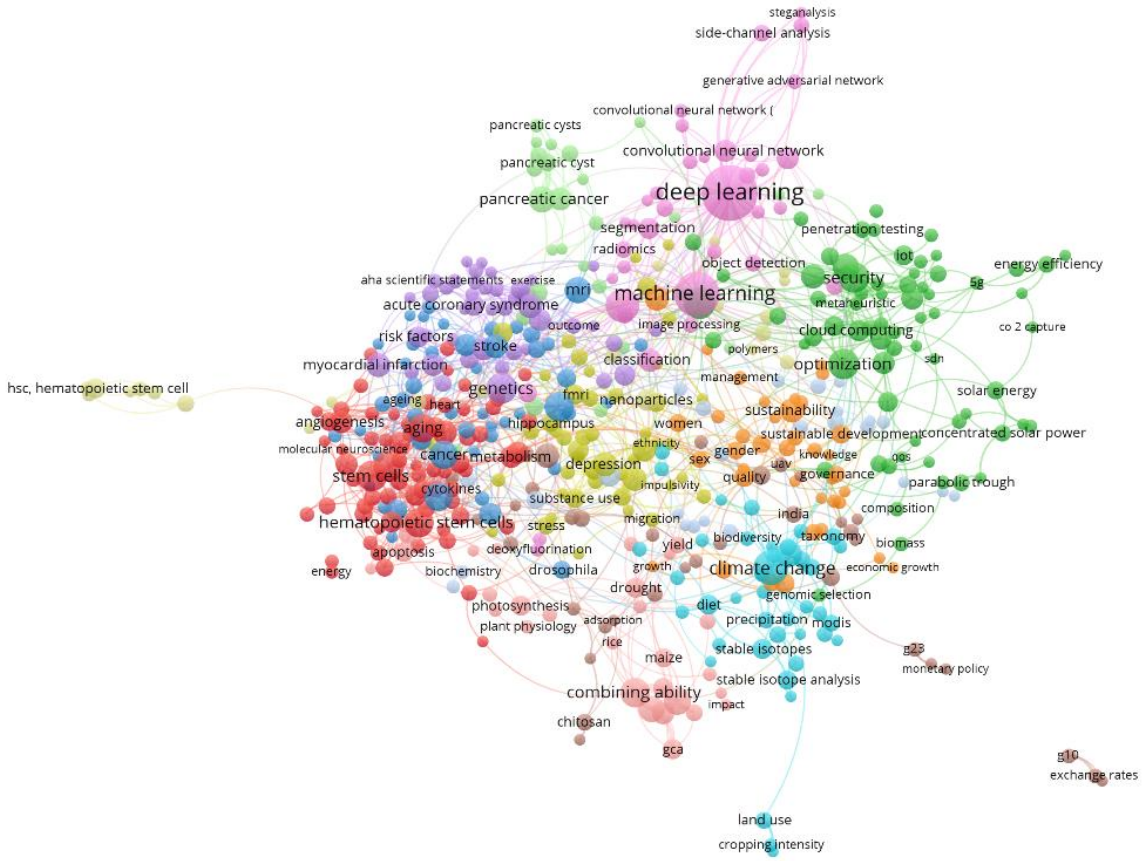

Fig 4. Co-occurrence map of keywords extracted from the analysed publications
Source: Authors’ elaboration based on VOSviewer.

Each node corresponds to a term or keyword (normalized in English), and the size of the node reflects the frequency of occurrence. Nodes are linked when terms appear together in a significant number of articles, suggesting a thematic relationship. Colors indicate clusters of correlated terms. We can observe a main cluster (green) centered on web vulnerabilities – encompassing terms such as “SQL injection”, “XSS”, “web security”, “input validation” –, and another cluster (yellow) focused on automated analysis techniques – including “static code analysis”, “machine learning”, “vulnerability scanners”, “fuzzing”. There are also smaller subclusters: for example, one related to security in .NET architecture (terms such as “authentication”, “ASP.NET”, “IIS configuration”) and another linked to machine learning (terms such as “neural networks”, “data set”, possibly connected with vulnerability detection

through deep learning). This map shows that the literature combines traditional application security concerns with new automation and artificial intelligence techniques.

Figure 4 confirms that SQL Injection and XSS are among the most central topics (largest nodes in the green cluster), which is not surprising given the historical prevalence of these vulnerabilities and the focus on automating their detection/remediation. The proximity of "web application" and "OWASP Top 10" to these nodes, suggests that many studies structure their contributions around the failure categories highlighted by OWASP (OWASP, 2023). In the yellow cluster, we see "static analysis" strongly linked to "vulnerability detection", indicating the popularity of automated static analysis tools in the corpus. "Machine learning" appears connected both to "vulnerability detection" and to terms such as "code gadgets" and "feature extraction" (terms specific to ML techniques applied to code), revealing that a notable portion of recent work explores learning algorithms to identify vulnerability patterns in source or binary code (Bavota et al., 2020). However, it is worth noting the relative absence of terms explicitly linked to .NET in the main clusters – for example, "C#" or ".NET Core" did not emerge as relevant nodes on the map. Only the term "ASP.NET" appears in a subcluster, associated with web application case studies. This may indicate that many searches are generic in nature (applicable to multiple platforms) or that, when they refer to .NET, they implicitly treat it in the web context. This finding already suggests a possible gap: a lack of work focused on the .NET platform as such, with studies that address web vulnerability cross-sectionally or analysis techniques that could be applied to .NET but are not discussed in the specific context of the platform being more common.

Other descriptive results: Regarding publication venues, the results include a combination of security conferences (e.g., ACM CCS, IEEE S&P in some cases for advanced techniques), software engineering conferences with security tracks (e.g., IEEE/ACM ASE, SANER), and specialized security or software journals (such as Computers & Security, IEEE Transactions on Reliability, Empirical Software Engineering, and more recently ACM Computing Surveys for the systematic reviews). Some regional journals or those with a lower impact factor also appear – particularly in the more applied works, such as the examples by Kausar et al. and Sharma & Sheth, published in magazines focused on practical technology. This suggests that the topic has attracted both cutting-edge research (published in major outlets) and more local/practical contributions. The geographic distribution of authors, inferred from affiliations, indicates a significant participation of authors from South and East Asia (India, China) in the most practical work on ASP.NET, while many of the machine learning approaches

came from groups in Europe and North America. This distribution reinforces the idea of subcommunities: one more oriented towards solving immediate problems in web application security (often in local contexts), and another more oriented towards broad methodological proposals (such as AI techniques) coming from established research centers.

Table 2 highlights the diversity of venues where studies were published, including multidisciplinary journals and technical conferences, reflecting the transversal nature of security research and automated software analysis.

Table 2. Diversity of venues where studies were published, including multidisciplinary journals and technical conferences

Source	Number of publications
Scientific Reports	159
PLOS ONE	110
Critical Care	94
SSRN Electronic Journal	83
HortScience	79
IEEE Access	77
Sensors	69
Remote Sensing	66
Nature Communications	61

Source: Authors' elaboration based on data form The Lens.

In short, the bibliometric results demonstrate a growing and diversifying field. There is evident interest in automating the security of .NET applications, focused mainly on preventing known web vulnerabilities and incorporating new techniques (AI, advanced analysis) to improve detection. However, signs of fragmentation and possible underexplored areas also emerge. In the following section, we critically discuss these results, with an emphasis on identified gaps and future trends.

5. Discussion

Bibliometric analysis revealed not only an increased interest in automated application security, but also significant thematic fragmentation and surprising selectivity in scientific literature. While there are relatively abundant publications on vulnerabilities in web applications developed with ASP.NET, other important domains of the .NET platform, such as

desktop applications (WPF, WinForms) and cross-platform mobile development (e.g., .NET MAUI), are almost entirely absent from peer-reviewed scientific publications.

This absence is particularly relevant considering that .NET desktop applications continue to be widely used in enterprise environments, especially in internal tools and legacy systems (Köse & Yilmaz, 2022). Similarly, with the release of .NET MAUI as a unified framework for mobile and desktop development, one would expect an increase in the number of studies focusing on security in this context. However, the analysis revealed virtually no publications dedicated to vulnerability detection or best practices for secure development in these areas.

Several reasons may justify this scenario. First, the scientific community tends to prioritize web security because it is more exposed and related to cloud-based or internet-oriented systems. Desktop applications are often seen as local or legacy systems, whose security depends more on the operating system than on the code itself (Salem et al., 2021). However, this view ignores important risks such as DLL injection, privilege escalation, and local data leaks—common in poorly configured or insecure software.

Second, development with .NET MAUI is a recent trend, and scientific literature may still be adapting. However, other mobile platforms, such as Android or Flutter, already have a robust body of studies on security and automated tools (Zhou et al., 2012), highlighting a missed opportunity for the .NET community to keep up with this movement.

Another critical point observed is the scarcity of empirical evaluations of widely used industry tools such as SonarQube, Fortify, or CodeQL. Few studies in the analyzed sample compare or evaluate the real-world performance of these tools on .NET codebases, suggesting a disconnect between academic research and professional practice. Recent studies reinforce this criticism, pointing to the need for more empirical validation and real-world case studies with automated security tools (Bavota et al., 2020).

Furthermore, the absence of platform-specific technical keywords, such as “C#”, “IL analysis”, or “.NET Core security model”, suggests that, even when .NET is addressed, many studies treat security generically, without attention to the architectural particularities of the framework. This compromises the applicability of many proposals, which may not be suitable for the characteristics of the runtime, memory management, or dependency injection practices of .NET.

In short, the results reinforce the need for greater alignment between the evolution of the .NET ecosystem and automated security research. The lack of studies focused on desktop and mobile contexts, the scarcity of empirical analyses of real-world tools, and the

superficial treatment of the framework's specificities represent concrete knowledge gaps. Overcoming them requires greater collaboration between academia, industry, and the .NET technical community itself.

6. Conclusions

This bibliometric review mapped the state of the art on automated security and vulnerability analysis in applications developed with the .NET framework. The results show that, although there is growing scientific interest in the topic, the field is still fragmented, with significant thematic gaps and low articulation between research groups. Most studies focus on ASP.NET-enabled web applications, while security in desktop (.NET Framework, WPF) and mobile (.NET MAUI) applications remains underexplored.

In addition to its academic contribution, this mapping offers direct Insights for professional practice. By identifying the most recurrent approaches, such as static analysis (SAST), the use of machine learning for vulnerability detection, and integration with DevSecOps pipelines, the study helps professionals understand which practices and tools are most prevalent and validated by literature. This makes it possible to make more informed choices when adopting security automation solutions.

For example, the strong presence of studies on static analysis tools suggests that integrating solutions like SonarQube, Fortify, or CodeQL into development workflows can bring concrete benefits in the early detection of critical flaws, such as SQL Injection or XSS—especially relevant in systems developed with ASP.NET. Similarly, the growing interest in machine learning models applied to code analysis points to an emerging area that can be explored by teams with DevOps maturity and availability of internal data.

Furthermore, the analysis revealed a scarcity of comparative studies between the tools, which reinforces the importance of evaluating each solution in the organization's real-world context. It is recommended that development professionals and security architects conduct pilot tests and benchmarks with their own .NET codebases, especially in less documented environments (such as desktop applications or MAUI), where generic tools may not be as effective.

Finally, the observed fragmentation suggests the need for closer collaboration between academia and the productive sector, so that research efforts are better aligned with the concrete

challenges faced by development teams. This is particularly relevant in organizations where security automation has not yet been systematically incorporated.

References

Aria, M., & Cuccurullo, C. (2017). bibliometrix: An R-tool for comprehensive science mapping analysis. *Journal of Informetrics, 11*(4), 959–975. https://doi.org/10.1016/j.joi.2017.08.007

Bavota, G., Russo, B., Oliveto, R., & Canfora, G. (2020). Empirical research in software security: Trends and gaps. *Empirical Software Engineering, 25*(6), 4393–4435.

Donthu, N., Kumar, S., Mukherjee, D., Pandey, N., & Lim, W. M. (2021). How to conduct a bibliometric analysis: An overview and guidelines. *Journal of Business Research, 133*, 285–296. https://doi.org/10.1016/j.jbusres.2021.04.070

Harzevili, N. S., Boaye Belle, A., Wang, J., Wang, S., Jiang, Z. M. (Jack), & Nagappan, N. (2024). A systematic literature review on automated software vulnerability detection using machine learning. *ACM Computing Surveys, 57*(3), Article 55, 1–36. https://doi.org/10.1145/3699711

Kausar, M. A., Nasar, M., & Moyaid, A. (2019). SQL injection detection and prevention techniques in ASP.NET web application. *International Journal of Recent Technology and Engineering, 8*(3), 7759–7766. https://doi.org/10.35940/ijrte.C6319.098319

Kausar, S., et al. (2019). A comparative study of static and dynamic analysis tools for detecting vulnerabilities in .NET applications. *Oriental Journal of Computer Science & Technology, 12*(1), 45–52.

Köse, U., & Yilmaz, R. M. (2022). Current and future trends in desktop application development: A review. *Journal of Software Engineering and Applications, 15*(1), 1–18.

OWASP. (2023). *OWASP Top Ten 2023*. OWASP Foundation. https://owasp.org/www-project-top-ten/

Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., … Moher, D. (2021). The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ (Clinical Research Ed.), 372*, n71. https://doi.org/10.1136/bmj.n71

PRISMA statement. (no date). PRISMA Statement. Retrieved December 3, 2025, from https://www.prisma-statement.org/

Salem, M. B., Hossain, M. S., & Zahedi, M. (2021). Security issues in desktop applications: A review. *International Journal of Computer Applications, 183*(28), 1–8.

Sharma, A., & Sheth, A. (2017). Towards practical vulnerability detection in modern web applications. In *Proceedings of the ACM Conference on Security and Privacy in Wireless and Mobile Networks*.

Sharma, R., & Sheth, R. (2017). Secure ASP.NET web application by discovering broken authentication and session management vulnerabilities. *Oriental Journal of Computer Science & Technology, 10*(2), 359–363. https://doi.org/10.13005/ojcst/10.02.15

van Eck, N. J., & Waltman, L. (2010). Software survey: VOSviewer, a computer program for bibliometric mapping. *Scientometrics, 84*(2), 523–538. https://doi.org/10.1007/s11192-009-0146-3

Williams, J., & Wichers, D. (2021). *DevSecOps and software supply chain security*. OWASP Foundation.

Zhou, Y., Zhang, X., & Jiang, X. (2012). Dissecting Android malware: Characterization and evolution. In *2012 IEEE Symposium on Security and Privacy* (pp. 95–109). IEEE. https://doi.org/10.1109/SP.2012.16

Zupic, I., & Čater, T. (2015). Bibliometric methods in management and organization. *Organizational Research Methods, 18*(3), 429–472.